

MS-VIEWSS Test Suite Specification

**Abstract:** This document provides information about how to configure the test suite and how MS-VIEWSS test suite is designed to test the MS-VIEWSS Open Specification usability and accuracy. It describes test assumptions, scope and constraints of the test suite. It also specifies test scenarios, detailed test cases, test suite architecture and adapter design**.**

Contents

[1 Configuring the test suite 3](#_Toc388454841)

[1.1 Configuring the test suite client 3](#_Toc388454842)

[1.1.1 Configuring the test suite client manually 3](#_Toc388454843)

[1.1.2 Configuring the test suite client by scripts 4](#_Toc388454844)

[1.2 Configuring the system under test (SUT) 4](#_Toc388454845)

[1.2.1 Configuring the SUT manually 4](#_Toc388454846)

[1.2.2 Configuring the SUT by scripts 4](#_Toc388454847)

[1.3 Configuring the SHOULD/MAY requirements 4](#_Toc388454848)

[2 Test suite design 6](#_Toc388454849)

[2.1 Assumptions, scope and constraints 6](#_Toc388454850)

[2.2 Test suite architecture 6](#_Toc388454851)

[2.3 Technical dependencies and considerations 8](#_Toc388454852)

[2.4 Adapter design 8](#_Toc388454853)

[2.4.1 Adapter overview 8](#_Toc388454854)

[2.4.2 Technical feasibility of adapter approach 9](#_Toc388454855)

[2.4.3 Adapter abstract layer 9](#_Toc388454856)

[2.4.4 Adapter details 9](#_Toc388454857)

[2.5 Test scenarios 11](#_Toc388454858)

[2.5.1 S01\_AddDeleteViews 12](#_Toc388454859)

[2.5.2 S02\_GetAllViews 12](#_Toc388454860)

[2.5.3 S03\_MaintainViewDefinition 12](#_Toc388454861)

[2.5.4 S04\_MaintainViewProperties 13](#_Toc388454862)

[2.5.5 S05\_MaintainViewPropertiesWithOpenApplicationExtension 13](#_Toc388454863)

[2.6 Test case design 13](#_Toc388454864)

[2.6.1 Traditional test case design 13](#_Toc388454865)

[2.6.2 Test case description 17](#_Toc388454866)

# Configuring the test suite

## Configuring the test suite client

### Configuring the test suite client manually

Before you run the test suite, update the values in the MS-VIEWSS\_TestSuite.deployment.ptfconfig file. The MS-VIEWSS\_TestSuite.deployment.ptfconfig file can also be configured by running the client setup script.

1. Open MS-VIEWSS\TestSuite\MS-VIEWSS\_TestSuite.deployment.ptfconfig.
2. Update the following value to specify the common configuration file.

Property name="CommonConfigurationFileName" value="SharePointCommonConfiguration.deployment.ptfconfig"

**Note** This property can be removed or set to empty if the required properties are copied to the test suite specific configuration file. Any other change to this property causes all test cases in the test suite to fail during execution. The test suite searches through its specific configuration file and uses those properties, if they are defined, before looking for them in the common configuration file (if specified).

1. Update the following properties' values to match SUT settings and configuration.

* Property name="TargetServiceUrl" value="[TransportType]://[SUTComputerName]/sites/[SiteCollectionName]/\_vti\_bin/views.asmx"
* Property name="SiteCollectionName" value="MSVIEWSS\_SiteCollection"
* Property name="ListsServiceUrl" value="[TransportType]://[SUTComputerName]/sites/[SiteCollectionName]/\_vti\_bin/lists.asmx"
* Property name="DisplayListName" value="MSVIEWSS\_ViewList"
* Property name="AllItemsCount" value="8"
* Property name="IsRowPaged" value="TRUE"
* Property name="AvailableRowLimit" value="120"
* Property name="OpenApplicationExtension" value=".doc"
* Property name="ViewFields0" value="ID"
* Property name="ViewFields1" value="Title"
* Property name="ViewFields2" value="Modified"
* Property name="ViewFields3" value="Modified By"
* Property name="FieldRefWhere\_Name" value="ID"
* Property name="FieldRefWhere\_Type" value="String"
* Property name="FieldRefWhere\_Text" value="2"
* Property name="QueryItemsCount" value="1"
* Property name="FieldRefOrderBy\_Name" value="ID"
* Property name="FieldRefOrderBy\_Ascending" value="FALSE"
* Property name="FieldRefGroupBy\_Name" value="Title"
* Property name="FieldRefGroupBy\_RowLimit" value="5"
* Property name="FieldRefGroupBy\_Ascending" value="FALSE"
* Property name="FieldRefAggregations\_Name" value="ID"
* Property name="FieldRefAggregations\_AggregationsType" value="Count"
* Property name="FormatsField\_Name" value="ID"
* Property name="FormatsField\_Type" value="RowHeight"
* Property name="FormatsField\_Value" value="30"
* Property name="FormatsGeneral\_Type" value="ColWidth"
* Property name="FormatsGeneral\_Value" value="30"

1. The following property is not associated with SUT settings and can normally retain with the default values.

* Property name="ServiceTimeOut" value="10"

### Configuring the test suite client by scripts

To configure the test suite client using scripts, see section 5.2.4 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx).

## Configuring the system under test (SUT)

### Configuring the SUT manually

To manually configure the SUT, see section 5.1.3 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx).

### Configuring the SUT by scripts

To configure the SUT using scripts, see section 5.1.2 of the [SharePointTestSuiteDeploymentGuide.docx](../SharePointTestSuiteDeploymentGuide.docx).

## Configuring the SHOULD/MAY requirements

Implementation of the SHOULD/MAY and endnote-related requirements are pre-configured in the format "<Property name="RXXXEnabled" value="VIEWSS"/>" for the product Views in the following config files:

* MS-VIEWSS\_WindowsSharePointServices3\_SHOULDMAY.deployment.ptfconfig
* MS-VIEWSS\_SharePointServer2007\_SHOULDMAY.deployment.ptfconfig
* MS-VIEWSS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig
* MS-VIEWSS\_SharePointServer2010\_SHOULDMAY.deployment.ptfconfig
* MS-VIEWSS\_SharePointFoundation2013\_SHOULDMAY.deployment.ptfconfig
* MS-VIEWSS\_SharePointServer2013\_SHOULDMAY.deployment.ptfconfig

If RXXXEnabled is set to true, the requirement must be checked. If false, the requirement must not be checked. For Microsoft product views, all values should not be changed. For third-party products, the closest Microsoft product version should be chosen, and the value of RXXXEnabled should be updated according to the real product behavior. For example, if SharePoint Foundation 2010 is chosen,user can open **MS-VIEWSS\_SharePointFoundation2010\_SHOULDMAY.deployment.ptfconfig** and update the RXXXEnabled accordingly.

# Test suite design

## Assumptions, scope and constraints

Assumptions

None.

Scope

In scope

* This test suite verifies the accuracy and integrity of the technical content in the Open Specification against the results returned from the protocol server by using eight operations: AddView**,** DeleteView**,** GetView, GetViewCollection, GetViewHtml, UpdateView, UpdateViewHtml,andUpdateViewHtml2**.**
* This test suite verifies the Full WSDL which is provided in the Open Specification.
* This test suite verifies the server-side and testable requirements by running all the test cases on both HTTP and HTTPS.
* This test suite verifies the requirements of the XML schema definition in MS-WSSCAML that is used by the typical scenarios of the MS-VIEWSS Open Specification. The in-scope MS-WSSCAML requirements are gathered in MS-VIEWSS requirement specification.

Out of scope

* This test suite does not verify the requirements related to client behaviors.
* This test suite does not verify the requirements related to server internal behaviors.
* This test suite does not verify the internal implementations of its transport protocol stack.
* This test suite does not verify the in-depth and detailed requirements of MS-WSSCAML that are not gathered in MS-VIEWSS requirement specification.

Constraints

None.

## Test suite architecture

This test suite verifies the server-side and testable requirements obtained from the Open Specification. The following figure shows the architecture of this test suite.



MS-VIEWSS test suite architecture

The details of the MS-VIEWSS test suite architecture

* SUT hosts the Views Web Service which this test suite runs against.
* From third-party user’s point of view, the SUT is the protocol server implementation.
* The following products have been tested with the MS-VIEWSS test suite on the Windows platform.
* Windows SharePoint Service 3.0 SP3
* Microsoft Office SharePoint Server 2007 SP3
* Microsoft SharePoint Foundation 2010 SP1
* Microsoft SharePoint Server 2010 SP1
* Microsoft SharePoint Foundation 2013 SP1
* Microsoft SharePoint Server 2013 SP1
* The test suite acts as the client to communicate with the SUT and validates the requirements gathered from the MS-VIEWSS Open Specification.
* Test cases use the MS-VIEWSS adapter to call and get the results of the MS-VIEWSS operations. Test cases also use the SUT control adapter to retrieve information from the SUT.
* MS-VIEWSS adapter is used in the test cases. The test cases call the methods in the interface to invoke the MS-VIEWSS protocol adapter’s operations.
* SUT control adapter is used in the test cases. The test cases call the methods in the interface to retrieve information from the SUT.

## Technical dependencies and considerations

Dependencies

* This test suite depends on the SOAP messaging protocol for exchanging structured data and type information.
* This test suite depends on the HTTP protocol or HTTPS protocol to transmit the messages.
* This test suite depends on the wsdl.exe tool in the .NET Framework SDK to generate the MS-VIEWSS proxy class.
* This test suite depends on the Protocol Test Framework (PTF) to derive managed adapters.
* This test suite depends on the MS-LISTSWS methods to retrieve the list GUID and retrieve the items count in a list view from the SUT.

Encryption consideration

* Transportation of MS-VIEWSS includes HTTP and HTTPS, and encryption is handled by HTTPS.

## Adapter design

### Adapter overview

One protocol adapter and one SUT control adapter are used in this test suite.

Protocol adapter

* MS-VIEWSS protocol adapter
* The MS-VIEWSS adapter is a managed adapter, which derives from the ManagedAdapterBase class in PTF.
* The MS-VIEWSS adapter has the following functionalities
* Choose HTTP or HTTPS and SOAP 1.1 or 1.2 for transport;
* Construct requests of eight MS-VIEWSS operations;
* Communicate with the SUT by sending requests to the SUT and receive the corresponding responses from the SUT;
* Parse the response messages and validate the messages according to the WSDL schema;
* The MS-VIEWSS adapter uses the C# proxy class, which is generated by running the wsdl.exe tool against the full WSDL of this protocol to send SOAP request messages and receive SOAP response messages. The wsdl.exe can be found in Microsoft .NET Framework SDK tools.

SUT control adapters

* SUT control adapter
* The SUT control adapter is a managed adapter, which derives from the ManagedAdapterBase class in PTF.
* The SUT control adapter has the following functionalities
* Retrieve the GUID of the specified list on the SUT.
* Retrieve the count of the list items in the specified view on the SUT.
* Retrieve the name of the default view for the specified list on the SUT.
* The SUT control adapter is invoked by the test cases.

### Technical feasibility of adapter approach

Message generation

* The MS-VIEWSS adapter gets the parameter values of the WSDL operations and calls the corresponding operations in MS-VIEWSS proxy class, the MS-VIEWSS proxy class serializes the parameter values to XML elements to format the SOAP request messages, then the SOAP request messages are sent out by the MS-VIEWSS proxy class.

Message consumption

* The messages received from the SUT are parsed in the MS-VIEWSS proxy class and is passed upon to the MS-VIEWSS adapter. Then these messages are consumed in the MS-VIEWSS adapter to validate the message format and to validate the logic-related requirements in the test cases.

SUT control adapter

* The SUT control adapter is designed to retrieve the list and view information from the SUT.

### Adapter abstract layer

Protocol adapter

MS-VIEWSS adapter interface

There are eight methods declared in the MS-VIEWSS adapter interface IMS\_VIEWSSAdapter.

* These methods correspond to the eight MS-VIEWSS operations: AddView**,** DeleteView**,** GetView, GetViewCollection, GetViewHtml, UpdateView, UpdateViewHtml,andUpdateViewHtml2**.** The operators of these methods are abstracted the same as the operations specified in the MS-VIEWSS.

SUT control adapters

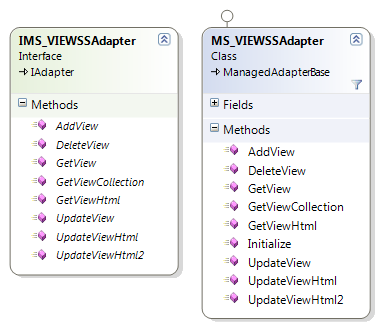
* SUT control adapter interface
* Three methods corresponding with the following three functions are declared in the SUT control adapter interface IMS\_VIEWSSSUTControlAdapter.
* Get the GUID of the specified list.
* Get the count of the list items in the specified view.
* Retrieve the name of the default view for the specified list on the SUT.

### Adapter details

#### Protocol adapter

##### MS-VIEWSS protocol adapter

The following figure shows the class diagram of the MS-VIEWSS protocol adapter.



MS-VIEWSS adapter class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_VIEWSSAdapter is the interface of the protocol adapter.
* IMS\_VIEWSSAdapter defines the methods invoked by test cases, including AddView**,** DeleteView**,** GetView, GetViewCollection, GetViewHtml, UpdateView, UpdateViewHtml,andUpdateViewHtml2 methods.

Adapter implementation

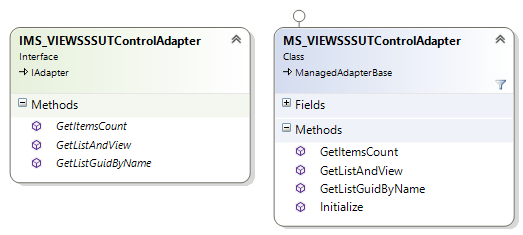
* MS\_VIEWSSAdapter is the protocol adapter class of the test suite. It is used to implement IMS\_VIEWSSAdapter.
* The Initialize method is used to initialize the MS-VIEWSS test suite.

#### SUT control adapter

The following outlines details of the class diagram:

##### SUT control adapter

The following figure shows the class diagram of the SUT Control Adapter.



SUT control adapter class diagram

The following outlines details of the class diagram:

Adapter interface

* IMS\_VIEWSSSUTControlAdapter is the interface of MS\_VIEWSSSUTControlAdapter.
* IMS\_VIEWSSSUTControlAdapter defines the methods invoked by test cases, including GetItemsCount, GetListAndView and GetListGuidByName methods.

Adapter implementation

* MS\_VIEWSSSUTControlAdapter is the SUT control adapter class of the test suite. It is used to implement IMS\_VIEWSSSUTControlAdapter.
* The Initialize method is used to initialize the SUT control adapter.
* The MS\_VIEWSSSUTControlAdapter is implemented by managed code. The implementation can be substituted by other implementation for the third party’s need.

## Test scenarios

Five scenarios are designed to cover the server-side, testable requirements in the MS-VIEWSS test suite. The details of the scenarios are as follows.

|  |  |
| --- | --- |
| Scenario | Description |
| S01\_AddDeleteViews | Add and delete a view for a specified list with valid or invalid input parameters. Get the added or deleted view for verification. |
| S02\_GetAllViews | Get all of the views of a specified list with valid or invalid input parameters. |
| S03\_MaintainViewDefinition | Get or update the definition of a view of a specified list with valid or invalid input parameters. |
| S04\_MaintainViewProperties | Get or update the definition and display properties of a view of a specified list with valid or invalid input parameters. |
| S05\_MaintainViewPropertiesWithOpenApplicationExtension | Get or update the definition and display properties of a view of a specified list with valid or invalid input parameters, with open application extension. |

MS-VIEWSS scenarios

### S01\_AddDeleteViews

Description

Add and delete a view for a specified list with valid or invalid input parameters. Get the added or deleted view for verification.

Operations

* AddView
* DeleteView

Prerequisites

* Get the list GUID from the SUT.

Cleanup

* Delete all the views that were added in the test case.

### S02\_GetAllViews

Description

Get all of the views of a specified list with valid or invalid input parameters.

Operations

* GetViewCollection

Prerequisites

* Get the list GUID from the SUT.

Cleanup

* Delete all the views that were added in the test case.

### S03\_MaintainViewDefinition

Description

Get or update the definition of a view of a specified list with valid or invalid input parameters.

Operations

* GetView
* UpdateView

Prerequisites

* Get the list GUID from the SUT.

Cleanup

* Delete all the views that were added in the test case.

### S04\_MaintainViewProperties

Description

Get or update the definition and display properties of a view of a specified list with valid or invalid input parameters.

Operations

* GetViewHtml
* UpdateViewHtml

Prerequisites

* Get the list GUID from the SUT.

Cleanup

* Delete all the views that were added in the test case.

### S05\_MaintainViewPropertiesWithOpenApplicationExtension

Description

Get or update the definition and display properties of a view of a specified list with valid or invalid input parameters, with open application extension.

Operations

* UpdateViewHtml2

Prerequisites

* Get the list GUID from the SUT.

Cleanup

* Delete all the views that were added in the test case.

## Test case design

### Traditional test case design

Traditional testing is adopted as the test approach for this test suite. The test cases are designed to cover the server-side and testable requirements.

There are 66 traditional test cases designed to cover the five scenarios mentioned in section [2.6 Test scenarios](#_Test_scenarios_2). Details of the traditional test cases are specified in section [2.7.2 Test case description](#_Test_case_description). The scenarios distributions of the test cases are listed in the following table.

|  |  |
| --- | --- |
| Scenario ID | Test case name |
| S01\_AddDeleteViews | MSVIEWSS\_S01\_TC01\_AddView\_Success |
| MSVIEWSS\_S01\_TC02\_AddView\_EmptyViewFields |
| MSVIEWSS\_S01\_TC03\_AddView\_EmptyQuery |
| MSVIEWSS\_S01\_TC04\_AddView\_EmptyRowLimit |
| MSVIEWSS\_S01\_TC05\_AddView\_EmptyRowLimitWithoutWhere |
| MSVIEWSS\_S01\_TC06\_AddView\_LogicalJoinDefinitionWithoutChild |
| MSVIEWSS\_S01\_TC07\_AddView\_LogicalJoinDefinitionPresent |
| MSVIEWSS\_S01\_TC08\_AddView\_NullRowLimitAndGroupByFalseCollapse |
| MSVIEWSS\_S01\_TC09\_AddView\_NullRowLimit |
| MSVIEWSS\_S01\_TC10\_AddView\_MakeViewDefaultTrue |
| MSVIEWSS\_S01\_TC11\_AddView\_NullType |
| MSVIEWSS\_S01\_TC12\_AddView\_EmptyType |
| MSVIEWSS\_S01\_TC13\_AddView\_InvalidType |
| MSVIEWSS\_S01\_TC14\_AddView\_InvalidListName |
| MSVIEWSS\_S01\_TC15\_AddView\_TrueCollapse\_NoComputedFields |
| MSVIEWSS\_S01\_TC16\_AddView\_WithoutOptionalParameters |
| MSVIEWSS\_S01\_TC17\_DeleteView\_Success |
| MSVIEWSS\_S01\_TC18\_DeleteView\_InvalidListName |
| MSVIEWSS\_S01\_TC19\_DeleteView\_WithoutOptionalParameters |
| MSVIEWSS\_S01\_TC20\_DeleteView\_NoDefaultView |
| S02\_GetAllViews | MSVIEWSS\_S02\_TC01\_GetViewCollection\_Success |
| MSVIEWSS\_S02\_TC02\_GetViewCollection\_InvalidListName |
| S03\_MaintainViewDefinition | MSVIEWSS\_S03\_TC01\_UpdateView\_InvalidListName |
| MSVIEWSS\_S03\_TC02\_UpdateView\_InvalidViewName |
| MSVIEWSS\_S03\_TC03\_GetView\_NullViewName |
| MSVIEWSS\_S03\_TC04\_GetView\_EmptyViewName |
| MSVIEWSS\_S03\_TC05\_GetView\_EmptyViewName\_NoDefaultView |
| MSVIEWSS\_S03\_TC06\_GetView\_Success |
| MSVIEWSS\_S03\_TC07\_UpdateView\_AllParameters |
| MSVIEWSS\_S03\_TC08\_UpdateView\_TrueCollapse\_NoComputedFields |
| MSVIEWSS\_S03\_TC09\_UpdateView\_LogicalJoinDefinitionPresent |
| MSVIEWSS\_S03\_TC10\_UpdateView\_LogicalJoinDefinitionWithoutChild |
| MSVIEWSS\_S03\_TC11\_UpdateView\_FalseCollapse |
| MSVIEWSS\_S03\_TC12\_GetView\_InvalidListName |
| MSVIEWSS\_S03\_TC13\_GetView\_InvalidViewName |
| MSVIEWSS\_S03\_TC14\_UpdateView\_NullViewName |
| MSVIEWSS\_S03\_TC15\_UpdateView\_EmptyViewName |
| MSVIEWSS\_S03\_TC16\_UpdateView\_EmptyViewName\_NoDefaultView |
| MSVIEWSS\_S03\_TC17\_UpdateView\_WithoutOptionalParameters |
| S04\_MaintainViewProperties | MSVIEWSS\_S04\_TC01\_GetViewHtml\_InvalidListName |
| MSVIEWSS\_S04\_TC02\_UpdateViewHtml\_InvalidListName |
| MSVIEWSS\_S04\_TC03\_GetViewHtml\_InvalidViewName |
| MSVIEWSS\_S04\_TC04\_UpdateViewHtml\_InvalidViewName |
| MSVIEWSS\_S04\_TC05\_GetViewHtml\_NullViewName |
| MSVIEWSS\_S04\_TC06\_UpdateViewHtml\_NullViewName |
| MSVIEWSS\_S04\_TC07\_GetViewHtml\_EmptyViewName |
| MSVIEWSS\_S04\_TC08\_UpdateViewHtml\_EmptyViewName |
| MSVIEWSS\_S04\_TC09\_GetViewHtml\_EmptyViewName\_NoDefaultView |
| MSVIEWSS\_S04\_TC10\_UpdateViewHtml\_AllParameters |
| MSVIEWSS\_S04\_TC11\_UpdateViewHtml\_WithoutOptionalParameters |
| MSVIEWSS\_S04\_TC12\_CompareUpdateViewHtmlResult |
| MSVIEWSS\_S04\_TC13\_GetViewHtml\_Success |
| MSVIEWSS\_S04\_TC14\_UpdateViewHtml\_TrueCollapse\_NoComputedFields |
| MSVIEWSS\_S04\_TC15\_UpdateViewHtml\_LogicalJoinDefinitionPresent |
| MSVIEWSS\_S04\_TC16\_UpdateViewHtml\_LogicalJoinDefinitionWithoutChild |
| MSVIEWSS\_S04\_TC17\_UpdateViewHtml\_FalseCollapse |
| MSVIEWSS\_S04\_TC18\_UpdateViewHtml\_EmptyViewName\_NoDefaultView |
| S05\_MaintainViewPropertiesWithOpenApplicationExtension | MSVIEWSS\_S05\_TC01\_UpdateViewHtml2\_Success |
| MSVIEWSS\_S05\_TC02\_UpdateViewHtml2\_LogicalJoin |
| MSVIEWSS\_S05\_TC03\_UpdateViewHtml2\_GroupByAndCollapse |
| MSVIEWSS\_S05\_TC04\_UpdateViewHtml2\_NullViewName |
| MSVIEWSS\_S05\_TC05\_UpdateViewHtml2\_EmptyViewName |
| MSVIEWSS\_S05\_TC06\_UpdateViewHtml2\_Success\_LeastInputParameters |
| MSVIEWSS\_S05\_TC07\_UpdateViewHtml2\_InvalidListName |
| MSVIEWSS\_S05\_TC08\_UpdateViewHtml2\_InvalidViewName |
| MSVIEWSS\_S05\_TC09\_UpdateViewHtml2\_NoDefaultView |

Test case scenario distribution

Negative testing is used in all of the scenarios. The client sends invalid messages to the server; or the client sends correct messages to the server that is in a wrong state, expecting to get a SOAP fault message which is used to verify negative requirements as described in the Open Specification.

### Test case description

The following table describes common prerequisites and common cleanup for all the test cases:

|  |  |
| --- | --- |
| Common prerequisites | The client calls the SUT Control Adapter method GetListGuidByName to get the GUID of the list on the server. |
| Common cleanup | 1. Check what views were added in the test case. 2. For each view found in Step 1, the client calls the protocol adapter method DeleteView to delete it. 3. If the original default view lost its default view position, restore its default view position. |
| Test suite prerequisite | 1. Call the SUT control adapter method GetListAndView to check whether there is a default view on the list before any test case begins. 2. Remember the view name of the default view, if it exists. |

The steps in the following test cases use methods and parameters in the adapter interfaces directly.

The following tables describe the traditional test cases.

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC01\_AddView\_Success |
| **Description** | A test case used to test AddView method successfully with valid parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with valid parameters. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC01\_AddView\_Success

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC02\_AddView\_EmptyViewFields |
| **Description** | A test case used to test AddView method with an empty viewFields parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an empty viewFields. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC02\_AddView\_EmptyViewFields

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC03\_AddView\_EmptyQuery |
| **Description** | A test case used to test AddView method with an empty query parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an empty query. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC03\_AddView\_EmptyQuery

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC04\_AddView\_EmptyRowLimit |
| **Description** | A test case used to test AddView method with an empty rowLimit parameter and a query having Where condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an empty rowLimit and a query having the Where condition. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC04\_AddView\_EmptyRowLimit

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC05\_AddView\_EmptyRowLimitWithoutWhere |
| **Description** | A test case used to test AddView method with an empty rowLimit parameter and a query without Where condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an empty rowLimit a query without Where condition. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC05\_AddView\_EmptyRowLimitWithoutWhere

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC06\_AddView\_LogicalJoinDefinitionWithoutChild |
| **Description** | A test case used to test AddView method when there are no child elements in LogicalJoinDefinition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server when there are no child elements in LogicalJoinDefinition. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC06\_AddView\_LogicalJoinDefinitionWithoutChild

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC07\_AddView\_LogicalJoinDefinitionPresent |
| **Description** | A test case used to test AddView method when there are child elements in LogicalJoinDefinition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server when there are child elements in LogicalJoinDefinition. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC07\_AddView\_LogicalJoinDefinitionPresent

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC08\_AddView\_NullRowLimitAndGroupByFalseCollapse |
| **Description** | A test case used to test AddView method with null rowLimit parameter and collapse is set to false in the GroupBy query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with null rowLimit and a query having GroupBy condition and Collapse as false. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC08\_AddView\_NullRowLimitAndGroupByFalseCollapse

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC09\_AddView\_NullRowLimit |
| **Description** | A test case used to test AddView method with null rowLimit parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with null rowLimit. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC09\_AddView\_NullRowLimit

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC10\_AddView\_MakeViewDefaultTrue |
| **Description** | A test case used to test AddView method with makeViewDefault parameter is set true. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with makeViewDefault is set true. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC10\_AddView\_MakeViewDefaultTrue

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC11\_AddView\_NullType |
| **Description** | A test case used to test AddView method with null type parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with null type. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC11\_AddView\_NullType

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC12\_AddView\_EmptyType |
| **Description** | A test case used to test AddView method with an empty type parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an empty type. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC12\_AddView\_EmptyType

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC13\_AddView\_InvalidType |
| **Description** | A test case used to test AddView method with invalid type parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with invalid type, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC13\_AddView\_InvalidType

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC14\_AddView\_InvalidListName |
| **Description** | A test case used to test AddView method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view with an invalid listName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC14\_AddView\_InvalidListName

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC15\_AddView\_TrueCollapse\_NoComputedFields |
| **Description** | A test case used to test AddView method without computed fields in viewFields and collapse is set to true in the query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view without computed fields in viewFields and collapse is set to true in the query condition. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC15\_AddView\_TrueCollapse\_NoComputedFields

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC16\_AddView\_WithoutOptionalParameters |
| **Description** | A test case used to test AddView method without the optional parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView without the optional parameters to add a list view. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC16\_AddView\_WithoutOptionalParameters

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC17\_DeleteView\_Success |
| **Description** | A test case used to test DeleteView method successful with valid parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call the protocol adapter method DeleteView with valid parameters to delete the list view created above. 4. Call the protocol adapter method GetView to get the list view deleted above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC17\_DeleteView\_Success

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC18\_DeleteView\_InvalidListName |
| **Description** | A test case used to test DeleteView method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call the protocol adapter method DeleteView to delete the list view with an invalid listName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC18\_DeleteView\_InvalidListName

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC19\_DeleteView\_WithoutOptionalParameters |
| **Description** | A test case used to test DeleteView method without the optional parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView with a null viewName. 3. Call the protocol adapter method GetView to get the default list view deleted above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC19\_DeleteView\_WithoutOptionalParameters

|  |  |
| --- | --- |
| **S01\_AddDeleteViews** | |
| **Test case ID** | MSVIEWSS\_S01\_TC20\_DeleteView\_NoDefaultView |
| **Description** | A test case used to test DeleteView operation to delete the default view when there is no default view in the list. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView to delete the default list view. 3. Call the protocol adapter method DeleteView method to delete the default view with an empty view name, when there is no default view in the list, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S01\_TC20\_DeleteView\_NoDefaultView

|  |  |
| --- | --- |
| **S02\_GetAllViews** | |
| **Test case ID** | MSVIEWSS\_S02\_TC01\_GetViewCollection\_Success |
| **Description** | A test case used to test GetViewCollection method successfully with valid parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView twice to add two views for the specified list on the server. 2. Call the protocol adapter method GetViewCollection to retrieve the collection of list views of a specified list in the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S02\_TC01\_GetViewCollection\_Success

|  |  |
| --- | --- |
| **S02\_GetAllViews** | |
| **Test case ID** | MSVIEWSS\_S02\_TC02\_GetViewCollection\_InvalidListName |
| **Description** | A test case used to test GetViewCollection method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method GetViewCollection with invalid listName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S02\_TC02\_GetViewCollection\_InvalidListName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC01\_UpdateView\_InvalidListName |
| **Description** | A test case used to test UpdateView method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetView to get the list view created above. 3. Call the protocol adapter method UpdateView to update the list view added above with an invalid listName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC01\_UpdateView\_InvalidListName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC02\_UpdateView\_InvalidViewName |
| **Description** | A test case used to test UpdateView method with invalid viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method UpdateView with an invalid viewName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC02\_UpdateView\_InvalidViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC03\_GetView\_NullViewName |
| **Description** | A test case used to test GetView method with null viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. C the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method GetView with a null viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC03\_GetView\_NullViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC04\_GetView\_EmptyViewName |
| **Description** | A test case used to test GetView method with an empty viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method GetView with an empty viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC04\_GetView\_EmptyViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC05\_GetView\_EmptyViewName\_NoDefaultView |
| **Description** | A test case used to test GetView method with an empty viewName parameter when the default list view does not exist. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView to delete the default list view. 3. Call the protocol adapter method GetView with an empty viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC05\_GetView\_EmptyViewName\_NoDefaultView

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC06\_GetView\_Success |
| **Description** | A test case used to test GetView method successful with valid parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetView to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC06\_GetView\_Success

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC07\_UpdateView\_AllParameters |
| **Description** | A test case used to test UpdateView method to update the FPModified field successfully with all input parameters present. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the FPModified field, with all input parameters present. 3. Call the protocol adapter method GetView to get the list view updated above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC07\_UpdateView\_AllParameters

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC08\_UpdateView\_TrueCollapse\_NoComputedFields |
| **Description** | A test case used to test UpdateView method without computed fields and collapse is set to true in the query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the display name of the list view created above with a query having GroupBy condition and Collapse as true, when there aren't any computed fields in the ViewFields section. 3. Call the protocol adapter method GetView to get the list view updated above. 4. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC08\_UpdateView\_TrueCollapse\_NoComputedFields

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC09\_UpdateView\_LogicalJoinDefinitionPresent |
| **Description** | A test case used to test UpdateView method when LogicalJoinDefinition is present and not empty in the query. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the query condition when LogicalJoinDefinition is present in the query and its child element is "Eq". 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC09\_UpdateView\_LogicalJoinDefinitionPresent

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC10\_UpdateView\_LogicalJoinDefinitionWithoutChild |
| **Description** | A test case used to test UpdateView method when there are no child elements in LogicalJoinDefinition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the query condition when there are no child elements in LogicalJoinDefinition. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC10\_UpdateView\_LogicalJoinDefinitionWithoutChild

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC11\_UpdateView\_FalseCollapse |
| **Description** | A test case used to test UpdateView method when collapse is set to false in the query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the list view to make it as default view. 3. Call the protocol adapter method GetView to get the list view updated above. 4. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC11\_UpdateView\_FalseCollapse

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC12\_GetView\_InvalidListName |
| **Description** | A test case used to test GetView method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetView with invalid listName to get the list view added above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC12\_GetView\_InvalidListName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC13\_GetView\_InvalidViewName |
| **Description** | A test case used to test GetView method with invalid viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method GetView with invalid viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC13\_GetView\_InvalidViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC14\_UpdateView\_NullViewName |
| **Description** | A test case used to test UpdateView method with null viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateView with a null viewName, expecting to update the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC14\_UpdateView\_NullViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC15\_UpdateView\_EmptyViewName |
| **Description** | A test case used to test UpdateView method with an empty viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateView to update the display name of the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC15\_UpdateView\_EmptyViewName

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC16\_UpdateView\_EmptyViewName\_NoDefaultView |
| **Description** | A test case used to test UpdateView method with an empty viewName parameter when the default list view does not exist. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView to delete the default view. 3. Call the protocol adapter method UpdateView to update the display name of the view with an empty viewName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC16\_UpdateView\_EmptyViewName\_NoDefaultView

|  |  |
| --- | --- |
| **S03\_MaintainViewDefinition** | |
| **Test case ID** | MSVIEWSS\_S03\_TC17\_UpdateView\_WithoutOptionalParameters |
| **Description** | A test case used to test UpdateView method without the optional parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateView without optional parameters to get the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S03\_TC17\_UpdateView\_WithoutOptionalParameters

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC01\_GetViewHtml\_InvalidListName |
| **Description** | A test case used to test GetViewHtml method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetViewHtml with invalid listName to get the list view added above, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC01\_GetViewHtml\_InvalidListName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC02\_UpdateViewHtml\_InvalidListName |
| **Description** | A test case used to test UpdateViewHtml method with invalid listName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml with invalid listName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC02\_UpdateViewHtml\_InvalidListName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC03\_GetViewHtml\_InvalidViewName |
| **Description** | A test case used to test GetViewHtml method with invalid viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method GetViewHtml with invalid viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC03\_GetViewHtml\_InvalidViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC04\_UpdateViewHtml\_InvalidViewName |
| **Description** | A test case used to test UpdateViewHtml method with invalid viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method UpdateViewHtml with invalid viewName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC04\_UpdateViewHtml\_InvalidViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC05\_GetViewHtml\_NullViewName |
| **Description** | A test case used to test GetViewHtml method with null viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method GetViewHtml with a null viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC05\_GetViewHtml\_NullViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC06\_UpdateViewHtml\_NullViewName |
| **Description** | A test case used to test UpdateViewHtml method with null viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml with null viewName, expecting to update the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC06\_UpdateViewHtml\_NullViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC07\_GetViewHtml\_EmptyViewName |
| **Description** | A test case used to test the GetViewHtml method with an empty viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method GetViewHtml with an empty viewName. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC07\_GetViewHtml\_EmptyViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC08\_UpdateViewHtml\_EmptyViewName |
| **Description** | A test case used to test the UpdateViewHtml method with an empty viewName parameter. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the Scope of the view with an empty viewName, expecting to update the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC08\_UpdateViewHtml\_EmptyViewName

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC09\_GetViewHtml\_EmptyViewName\_NoDefaultView |
| **Description** | A test case used to test GetViewHtml method with an empty viewName parameter when the default list view does not exist. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView to delete the default view. 3. Call the protocol adapter method GetViewHtml with an empty viewName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC09\_GetViewHtml\_EmptyViewName\_NoDefaultView

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC10\_UpdateViewHtml\_AllParameters |
| **Description** | A test case used to test UpdateViewHtml method successful with all parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the FPModified, with all input parameters present. 3. Call the protocol adapter method GetViewHtml to get the list view updated above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC10\_UpdateViewHtml\_AllParameters

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC11\_UpdateViewHtml\_WithoutOptionalParameters |
| **Description** | A test case used to test UpdateViewHtml method successful without the optional parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml without the optional parameters to get the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC11\_UpdateViewHtml\_WithoutOptionalParameters

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC12\_CompareUpdateViewHtmlResult |
| **Description** | A test case used to verify UpdateViewHtml and UpdateViewHtml2 return values are the same when OpenApplicationExtension is not present for UpdateViewHtml2. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the display name of the view added above. 3. Call the protocol adapter method UpdateViewHtml2 to update the display name of the view when openApplicationExtension is set to null and all other parameters are same as above UpdateViewHtml method. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC12\_CompareUpdateViewHtmlResult

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC13\_GetViewHtml\_Success |
| **Description** | A test case used to test GetViewHtml method with valid parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method GetViewHtml to get the list view created above. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC13\_GetViewHtml\_Success

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC14\_UpdateViewHtml\_TrueCollapse\_NoComputedFields |
| **Description** | A test case used to test UpdateViewHtml method without computed fields and collapse is set to true in the query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the display name of the list view added above. 3. Call the protocol adapter method GetViewHtml to get the list view updated above. 4. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC14\_UpdateViewHtml\_TrueCollapse\_NoComputedFields

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC15\_UpdateViewHtml\_LogicalJoinDefinitionPresent |
| **Description** | A test case used to test UpdateViewHtml method when LogicalJoinDefinition is present and not empty in the query. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the query condition when LogicalJoinDefinition is present in the query and its child element is "Eq". 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC15\_UpdateViewHtml\_LogicalJoinDefinitionPresent

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC16\_UpdateViewHtml\_LogicalJoinDefinitionWithoutChild |
| **Description** | A test case used to test UpdateViewHtml method when there are no child elements in LogicalJoinDefinition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the query condition when there are no child elements in LogicalJoinDefinition. 3. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC16\_UpdateViewHtml\_LogicalJoinDefinitionWithoutChild

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC17\_UpdateViewHtml\_FalseCollapse |
| **Description** | A test case used to test UpdateViewHtml method when Collapse is set to false in the GroupBy query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a list view for the specified list on the server. 2. Call the protocol adapter method UpdateViewHtml to update the display name of the list view added above, when Collapse is set to false in the GroupBy query condition. 3. Call the protocol adapter method GetViewHtml get the list view updated above. 4. Call SUT control adapter method GetItemsCount to get the count of the list items in the specified view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC17\_UpdateViewHtml\_FalseCollapse

|  |  |
| --- | --- |
| **S04\_MaintainViewProperties** | |
| **Test case ID** | MSVIEWSS\_S04\_TC18\_UpdateViewHtml\_EmptyViewName\_NoDefaultView |
| **Description** | A test case used to test UpdateViewHtml method with an empty viewName parameter when the default list view does not exist. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default list view for the specified list on the server. 2. Call the protocol adapter method DeleteView to delete the default view. 3. Call the protocol adapter method UpdateViewHtml with an empty viewName, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S04\_TC18\_UpdateViewHtml\_EmptyViewName\_NoDefaultView

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC01\_UpdateViewHtml2\_Success |
| **Description** | A test case used to test UpdateViewHtml2 operation to update the display name of a view, with all input parameters present. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a view. 2. Call the protocol adapter method UpdateViewHtml2 to update the display name of the view, with all input parameters present. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC01\_UpdateViewHtml2\_Success

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC02\_UpdateViewHtml2\_LogicalJoin |
| **Description** | A test case used to test UpdateViewHtml2 operation concerning the LogicalJoin query condition. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a view. 2. Call the protocol adapter method UpdateViewHtml2 to update the view with available LogicalJoinDefinition query condition. 3. Call SUT control adapter method GetItemsCount to get the count of the items in the view, expecting this value equals the available query result. 4. Call the protocol adapter method UpdateViewHtml2 to update the view with an empty query condition. 5. Call SUT control adapter method GetItemsCount to get the count of the items in the view, expecting this value equals the count of all the items in the list. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC02\_UpdateViewHtml2\_LogicalJoin

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC03\_UpdateViewHtml2\_GroupByAndCollapse |
| **Description** | A test case used to test UpdateViewHtml2 operation concerning the GroupBy and Collapse query conditions. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a view. 2. Call the protocol adapter method UpdateViewHtml2 to update the view with a query having GroupBy condition and Collapse as false. 3. Call SUT control adapter method GetItemsCount to get the count of the items in the view, expecting the value equals to the count of all the items in the list. 4. Call the protocol adapter method UpdateViewHtml2 to update the view with a query having GroupBy condition and Collapse as true, while the referenced view fields have no computed fields. 5. Call SUT control adapter method GetItemsCount to get the count of the items in the view, expecting the value equals to the count of all the items in the list. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC03\_UpdateViewHtml2\_GroupByAndCollapse

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC04\_UpdateViewHtml2\_NullViewName |
| **Description** | A test case used to test UpdateViewHtml2 operation with view name not present. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default view. 2. Call the protocol adapter method UpdateViewHtml2 with view name not present, expecting to update the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC04\_UpdateViewHtml2\_NullViewName

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC05\_UpdateViewHtml2\_EmptyViewName |
| **Description** | A test case used to test UpdateViewHtml2 operation with an empty view name. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default view. 2. Call the protocol adapter method UpdateViewHtml2 an empty view name, expecting to update the default view. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC05\_UpdateViewHtml2\_EmptyViewName

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC06\_UpdateViewHtml2\_Success\_LeastInputParameters |
| **Description** | A test case used to test UpdateViewHtml2 operation with the least input parameters. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default view. 2. Call the protocol adapter method UpdateViewHtml2 to get the default view with display properties by giving only one input parameter, the list name. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC06\_UpdateViewHtml2\_Success\_LeastInputParameters

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC07\_UpdateViewHtml2\_InvalidListName |
| **Description** | A test case used to test UpdateViewHtml2 operation with invalid list name. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a view. 2. Call the protocol adapter method UpdateViewHtml2 with invalid list name, expecting a SOAP fault from the protocol server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC07\_UpdateViewHtml2\_InvalidListName

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC08\_UpdateViewHtml2\_InvalidViewName |
| **Description** | A test case used to test UpdateViewHtml2 operation with invalid view name. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method UpdateViewHtml2 with invalid view name, expecting a SOAP fault from the protocol server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC08\_UpdateViewHtml2\_InvalidViewName

|  |  |
| --- | --- |
| **S05\_MaintainViewPropertiesWithOpenApplicationExtension** | |
| **Test case ID** | MSVIEWSS\_S05\_TC09\_UpdateViewHtml2\_NoDefaultView |
| **Description** | A test case used to test UpdateViewHtml2 operation to update the default view when the default view does not exist. |
| **Prerequisites** | Common prerequisites |
| **Test execution steps** | 1. Call the protocol adapter method AddView to add a default view. 2. Call the protocol adapter method DeleteView to delete the default view. 3. Call the protocol adapter method UpdateViewHtml2 with an empty view name to update the default view, expecting a SOAP fault from the server. 4. Call the protocol adapter method UpdateViewHtml2 with view name not present to update the default view, expecting a SOAP fault from the server. |
| **Cleanup** | Common cleanup |

MSVIEWSS\_S05\_TC09\_UpdateViewHtml2\_NoDefaultView

# 